NAME

dwarf_expand_frame_instructions - expand frame instructions

LIBRARY

DWARF Access Library (libdwarf, -ldwarf)

SYNOPSIS

#include <libdwarf.h>

int
dwarf_expand_frame_instructions(*Dwarf_Cie cie, Dwarf_Ptr instructions, Dwarf_Unsigned len, Dwarf_Frame_Op **ret_ops, Dwarf_Signed *ret_opcnt, Dwarf_Error *error*);

DESCRIPTION

Function **dwarf_expand_frame_instructions**() translates DWARF frame instruction bytes into an array of *Dwarf_Frame_Op* descriptors.

Argument *cie* should reference the CIE descriptor associated with the instructions to be translated.

Arugment *instructions* should point to an array of frame instruction bytes, as returned by the functions dwarf_get_cie_info(3) or dwarf_get_fde_instr_bytes(3).

Argument *len* should specify the number of the frame instruction bytes to be translated.

Argument *ret_ops* should point to a location that will be set to a pointer to an array of translated *Dwarf_Frame_Op* descriptors.

Argument *ret_opcnt* should point to a location that will hold the total number of the returned descriptors.

If argument *err* is not NULL, it will be used to store error information in case of an error.

Memory Management

The memory area used for the descriptor array returned in argument *ret_ops* is allocated by DWARF Access Library (libdwarf, -ldwarf). Application code should use function dwarf_dealloc(3) with type DW_DLA_FRAME_BLOCK to free the memory area when the descriptor array is no longer needed.

RETURN VALUES

Function **dwarf_expand_frame_instructions**() returns DW_DLV_OK when it succeeds. In case of an error, it returns DW_DLV_ERROR and sets the argument *err*.

EXAMPLES

To retrieve and expand the frame instructions for a given FDE descriptor, use:

```
Dwarf_Dbg dbg;
Dwarf_Cie cie;
Dwarf_Fde fde;
Dwarf_Ptr fde_inst;
Dwarf_Unsigned fde_instlen;
Dwarf_Frame_Op *ops;
Dwarf_Signed opcnt;
Dwarf_Error de;

/* ... assuming ‘dbg‘ references a valid DWARF debugging context,
  ‘fde‘ references a valid FDE descriptor and ‘cie‘ holds the CIE
  descriptor associated with the FDE descriptor ... */

if (dwarf_get_fde_instr_bytes(fde, &fde_inst, &fde_instlen,
    &de) != DW_DLV_OK)
        errx(EXIT_FAILURE, "dwarf_get_fde_instr_bytes failed: %s",
            dwarf_errmsg(de));

if (dwarf_expand_frame_instructions(cie, fde_inst, fde_instlen,
    &ops, &opcnt, &de) != DW_DLV_OK)
        errx(EXIT_FAILURE,
            "dwarf_expand_frame_instructions failed: %s",
            dwarf_errmsg(de));

for (i = 0; i < opcnt; i++) {
        /* ... use ops[i] ... */
}

/* Free the memory area when no longer needed. */
dwarf_dealloc(dbg, ops, DW_DLA_FRAME_BLOCK);
```

ERRORS

Function **dwarf_expand_frame_instructions**() can fail with:

[DW_DLE_ARGUMENT] One of the arguments *cie*, *instructions*, *ret_ops* or *ret_opcnt* was NULL.

[DW_DLE_ARGUMENT] Argument *len* was 0.

[DW_DLE_MEMORY] An out of memory condition was encountered during the execution of this function.

[DW_DLE_FRAME_INSTR_EXEC_ERROR]
An unknown instruction was found in the instruction bytes provided in argument *instructions*.

SEE ALSO

dwarf(3), dwarf_frame_instructions_dealloc(3), dwarf_get_cie_index(3), dwarf_get_cie_info(3), dwarf_get_cie_of_fde(3), dwarf_get_fde_at_pc(3), dwarf_get_fde_info_for_all_regs(3), dwarf_get_fde_info_for_all_regs3(3), dwarf_get_fde_info_for_cfa_reg3(3), dwarf_get_fde_info_for_reg(3), dwarf_get_fde_info_for_reg3(3), dwarf_get_fde_instr_bytes(3), dwarf_get_fde_list(3), dwarf_get_fde_list_eh(3), dwarf_get_fde_n(3)